Gnosis AI

Ανέβασε ένα CSV. Ρώτησε οτιδήποτε. Πάρε απάντηση σε δευτερόλεπτα.

Full-stack · Συνομιλιακό Analytics · LangChain RAG · 6 Πάροχοι LLM

● **LIVE** gnosis-ai-chi.vercel.app

Next.js 14 | FastAPI | LangChain RAG | Python 3.11 | Docker

Ένα ατομικό, end-to-end project — από 7 συνεντεύξεις χρηστών έως ζωντανή ανάπτυξη σε παραγωγή. Ερωτήματα σε φυσική γλώσσα δρομολογούνται σε προ-ελεγμένα εργαλεία Python μέσω ενός επιπέδου LangChain RAG· το LLM δεν βλέπει ποτέ ωμά δεδομένα και δεν γράφει ποτέ κώδικα. Τα αποτελέσματα είναι ντετερμινιστικά, σε ροή (streaming) και επεξηγήσιμα.

~11K	6	13	150	65%	<1s
ΓΡΑΜΜΕΣ ΚΩΔΙΚΑ	ΠΑΡΟΧΟΙ LLM	ΕΡΓΑΛΕΙΑ ML	TESTS	ΚΑΛΥΨΗ	1ο TOKEN P90

Τι είναι αυτό το έγγραφο — και πώς να το διαβάσετε

Αυτό το έγγραφο είναι το πλήρες αρχείο σχεδιασμού, μηχανικής και προϊόντος του **Gnosis AI** — μιας πλατφόρμας συνομιλιακού analytics που χτίστηκε ατομικά, από άκρη σε άκρη, από έρευνα χρηστών έως ζωντανή ανάπτυξη σε παραγωγή.

Καλύπτει 12 ενότητες πάνω σε προϊοντική σκέψη, τεχνική αρχιτεκτονική και αποφάσεις μηχανικής. Οι τρεις παρακάτω διαδρομές σας οδηγούν στις ενότητες που ταιριάζουν περισσότερο στον ρόλο σας.

PATH A

Οπτική Recruiter / PM

Προϊοντική σκέψη, έρευνα χρηστών και λήψη αποφάσεων

≈15 λεπτά ανάγνωση

PATH B

Οπτική Τεχνικού Αξιολογητή

Αρχιτεκτονική, σχεδιασμός RAG και τεχνικά trade-offs

≈18 λεπτά ανάγνωση

PATH C

Σύντομη Επισκόπηση

Βασικό προϊόν, βασικά διαφοροποιητικά στοιχεία και τι χτίστηκε

≈7 λεπτά ανάγνωση

Ποιος το έχτισε

Το Gnosis AI σχεδιάστηκε, αναπτύχθηκε και κυκλοφόρησε *εντελώς ατομικά* — χωρίς ομάδα, χωρίς PM, χωρίς υποστήριξη design. Κάθε απόφαση σε αυτό το έγγραφο πάρθηκε από το ίδιο άτομο που έγραψε τον κώδικα. Η σκέψη πίσω από κάθε

αρχιτεκτονική και προϊοντική επιλογή τεκμηριώνεται μαζί με τον κώδικα που την υλοποιεί — γιατί σε αυτό το build, ήταν η ίδια απόφαση.

01 ΕΠΙΣΚΟΠΗΣΗ & ΚΙΝΗΤΡΑ ΤΟΥ PROJECT

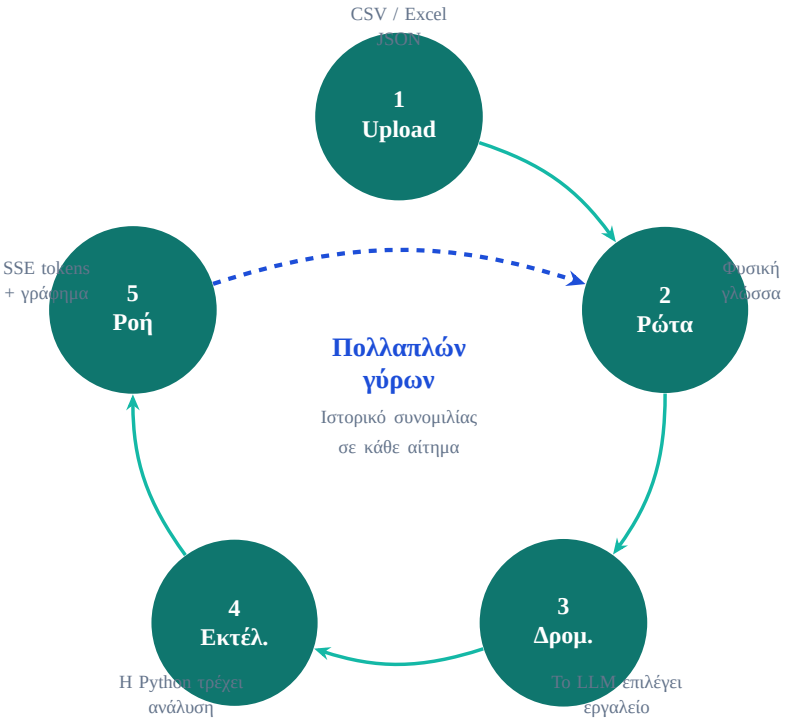
Γιατί το Έχτισα

Η εξερευνητική ανάλυση δεδομένων εξακολουθεί να απαιτεί ευχέρεια στην Python ή συνδρομή σε εργαλείο BI — ακόμα και όταν η ερώτηση που θέλει να κάνει ο χρήστης είναι συχνά απλή.

Οι αναλυτές δεδομένων ξοδεύουν δυσανάλογο χρόνο σε επαναλαμβανόμενες εξερευνητικές εργασίες: έλεγχοι ποιότητας δεδομένων, ανάλυση συσχέτισης, γρήγορη παλινδρόμηση, συνοπτικές αναφορές. Αυτά απαιτούν γνώση συγκεκριμένων εργαλείων, όχι εξειδικευμένη κρίση — είναι μηχανικά βήματα *πριν* ξεκινήσει η πραγματική ανάλυση. Το Gnosis AI εξαλείφει αυτή την επιβάρυνση: ανέβασε ένα CSV, ρώτησε σε φυσική γλώσσα, πάρε αποτέλεσμα.

Το project ήταν επίσης ένας σκόπιμος περιορισμός: να χτιστεί κάτι με πραγματική πολυπλοκότητα μηχανικής και προϊόντος — δρομολόγηση πολλαπλών LLM, streaming, πλήρη σουίτα tests — ως ατομικός developer υπό την ίδια πίεση πόρων και προτεραιοποίησης που αντιμετωπίζει ένας PM σε φάση 0-to-1. Κάθε απόφαση σε αυτό το έγγραφο αντανακλά αυτόν τον περιορισμό.

Ο Αναλυτικός Κύκλος



Πολλαπλών γύρων: κάθε απάντηση προτείνει την επόμενη ερώτηση μέσω context-aware chips — χωρίς κλήση στο backend.

02 ΕΡΕΥΝΑ ΧΡΗΣΤΩΝ & ΕΠΙΚΥΡΩΣΗ ΠΡΟΒΛΗΜΑΤΟΣ

> **Μέθοδος:** 7 ανεπίσημες ανιχνευτικές συζητήσεις — 4 διά ζώσης, 3 εξ αποστάσεως — πριν γραφτεί κανένας κώδικας παραγωγής. Οι συμμετέχοντες ήταν άνθρωποι που δουλεύουν με δεδομένα αλλά δεν είναι data scientists. Τους ζητήθηκε να περιγράψουν την τελευταία φορά που χρειάστηκε να κατανοήσουν ένα CSV γρήγορα. Χωρίς screener, χωρίς κίνητρο.

Τι Είπαν

Παράθεμα	Συμμετέχων	Προϊοντική επίπτωση
«Ξέρω τι θέλω να μάθω. Απλά δεν ξέρω πώς να το ζητήσω από το Excel. Ο πίνακας pivot μου πήρε 20 λεπτά και δεν είμαι σίγουρος ότι οι αριθμοί είναι σωστοί.»	P1 — Marketing Manager	Η πρόθεση είναι σαφής· το interface του εργαλείου είναι το μπλοκάρισμα. Η φυσική γλώσσα αφαιρεί το επίπεδο μετάφρασης.
«Το ChatGPT μου έδωσε έναν τοίχο κειμένου. Ήθελα απλά έναν αριθμό και ένα γράφημα. Κατέληξα να το κάνω στο Excel τελικά.»	P2 — Business Analyst	Τα γενικού σκοπού LLM παράγουν κείμενο, όχι ανάλυση. Ένα επίπεδο δρομολόγησης εργαλείων που τρέχει πραγματικούς υπολογισμούς είναι ένα θεμελιωδώς διαφορετικό προϊόν.
«Χρειάζομαι μια απάντηση τώρα, όχι ένα dashboard την επόμενη εβδομάδα.»	P6 — Product Manager	Το λανθάνον δεν είναι ποιότητα, είναι ταχύτητα. Τα SSE tokens ξεκινούν να εμφανίζονται σε λιγότερο από 1 δευτερόλεπτο.

03 JOBS TO BE DONE

Τρεις δουλειές, τρεις τρόποι αποτυχίας — προέκυψαν από 7 συνεντεύξεις, όχι υποθέσεις

	ΟΤΑΝ…	ΘΕΛΩ ΝΑ…	ΩΣΤΕ ΝΑ…
1	έχω ένα CSV με μια ερώτηση για την οποία γνωρίζω ήδη τη μορφή της απάντησης	παίρνω τον αριθμό (ή το γράφημα) σε λιγότερο από ένα λεπτό, χωρίς να γράψω τύπο ή να ανοίξω notebook	προχωρήσω — η ερώτηση δεν ήταν ποτέ το ενδιαφέρον κομμάτι, η απάντηση ήταν
2	ξέρω ποια ανάλυση χρειάζομαι (“Random Forest”, “clustering”) αλλά δεν μπορώ να την εκτελέσω	να την τρέξω με μια πρόταση, να δω feature importance και confidence του μοντέλου χωρίς να διαβάσω τεκμηρίωση	αποφασίσω αν το αποτέλεσμα είναι αρκετά καλό για να δράσω, ή να το παραδώσω σε κάποιον τεχνικό
3	χρειάζεται να δημοσιεύσω έναν αριθμό δημόσια και δεν μπορώ να κάνω λάθος	να δω ένα αναπαραγώγιμο ιχνηλάσιμο μονοπάτι — μέγεθος δείγματος, διαστήματα εμπιστοσύνης, μεθοδολογία — όχι μόνο μια απάντηση	να υποστηρίξω το αποτέλεσμα αν αμφισβητηθεί, χωρίς να έχω γράψει εγώ την ανάλυση

Δουλειά 1 · Απάντηση Gnosis

SSE streaming: πρώτο token σε <1s. Χωρίς λογαριασμό, χωρίς τύπο, χωρίς πίνακα pivot. Η απάντηση φτάνει πριν ο χρήστης ανοίξει το Excel.

Δουλειά 2 · Απάντηση Gnosis

Τα context-aware chips προτείνουν το `run_random_forest` με το όνομά του. Το panel επεξήγησης ML απαντά στο “γιατί αυτό το μοντέλο;” χωρίς να χρειάζεται ο χρήστης να ξέρει τι είναι το SHAP.

Δουλειά 3 · Απάντηση Gnosis

`generate_pdf_report`: N, διαστήματα εμπιστοσύνης, feature importance — διαμοιράσιμο χωρίς ο παραλήπτης να χρειάζεται πρόσβαση στην εφαρμογή.

04 3 ΑΡΧΕΤΥΠΑ ΧΡΗΣΤΩΝ

Μοτίβα συμπεριφοράς από 7 συνεντεύξεις — όχι δημογραφικά στοιχεία

Ο Κάτοχος της Ερώτησης · P1, P5, P6 · Marketing / Ops / PM

Ξέρει ακριβώς τι θέλει να μάθει. Μηδενικό ενδιαφέρον για εργαλεία, κώδικα ή μεθοδολογία. Λύνει το πρόβλημα με ανάθεση ή με χρονοβόρες λύσεις ανάγκης.

ΕΜΠΟΔΙΟ

Επιβάρυνση από το interface του εργαλείου. Ξέρει τι να ρωτήσει· δεν ξέρει ποιος τύπος του Excel το εξάγει.

ΕΝΕΡΓΟΠΟΙΗΣΗ

Γρήγορη απάντηση + γράφημα σε <60 δευτερόλεπτα. Υψηλό LTV — επιστρέφει με κάθε νέο dataset.

Ο Αναλυτής που Κόλλησε · P2, P3, P4 · BA / PhD / Founder

Έχει κάποια τεχνική εξοικείωση — άνοιξε ένα notebook, χρησιμοποίησε ChatGPT — αλλά σκαλώνει στην εκτέλεση. Ξέρει ότι “χρειάζομαι Random Forest” αλλά δεν μπορεί να το γράψει.

ΕΜΠΟΔΙΟ

Κενό εκτέλεσης. Ξέρει τον σωστό τύπο ανάλυσης· δεν μπορεί να τον τρέξει χωρίς βοήθεια.

ΕΝΕΡΓΟΠΟΙΗΣΗ

Πρώτη εκτέλεση εργαλείου ML. Context-aware chips με ονόματα αναλύσεων που ήδη γνωρίζει.

Ο Αναζητητής Σιγουριάς · P7, P3 · Journalist / Researcher

Μπορεί να παράγει ένα γράφημα (Datawrapper, Google Sheets). Το πρόβλημα δεν είναι η απάντηση — είναι η εμπιστοσύνη στην απάντηση. Δουλεύει δημόσια όπου ένας λάθος αριθμός έχει συνέπειες.

ΕΜΠΟΔΙΟ

Κενό εμπιστοσύνης. Παίρνει μια απάντηση· δεν μπορεί να την επαληθεύσει.

ΕΝΕΡΓΟΠΟΙΗΣΗ

Αναφορά PDF με N, διαστήματα εμπιστοσύνης, feature importance. Το πιο δύσκολο αρχέτυπο στη μετατροπή.

05 3 ΣΚΟΠΙΜΟΙ ΠΕΡΙΟΡΙΣΜΟΙ

Αρχιτεκτονικές αποφάσεις που αποκλείουν λειτουργικότητα σκόπιμα — και γιατί

ΠΡΟΚΛΗΣΗ Κάθε ανταγωνιστικό εργαλείο (Julius AI, ChatGPT ADA, Hex, Observable) απαιτεί λογαριασμό πριν ο χρήστης μπορέσει να κάνει οτιδήποτε. Αυτό είναι ένα εμπόδιο μετατροπής, ένας μηχανισμός συγκέντρωσης email, και ένας γνωστικός φόρος πριν ο χρήστης βιώσει οποιαδήποτε αξία.

ΛΥΣΗ ✓ Χωρίς Email. Χωρίς Εγγραφή. Χωρίς Λογαριασμό. Ένα Session UUID στο localStorage είναι η μόνη ταυτότητα που χρειάζεται το προϊόν. Ένας χρήστης που πέφτει σε τοίχο εγγραφής γυρίζει στο Excel. Όποιος παίρνει μια απάντηση σε 30 δευτερόλεπτα έχει μετατραπεί. *Trade-off που έγινε αποδεκτό*: χωρίς μόνιμους χώρους εργασίας, βαθμίδες χρέωσης ή διαχείριση ομάδας — σωστό scope για το v1.

ΠΡΟΚΛΗΣΗ Κάθε cloud-based ανταγωνιστής στέλνει δεδομένα χρήστη σε εξωτερικό API. Αυτό αποκλείει HR managers με δεδομένα προσωπικού, αναλυτές οικονομικών με μοντέλα πριν τα αποτελέσματα, ερευνητές υγείας, ομάδες legal υπό απαιτήσεις τοπικής φύλαξης δεδομένων.

ΛΥΣΗ ✓ Το Gnosis υποστηρίζει το Ollama ως πάροχο LLM πρώτης κατηγορίας. Με επιλεγμένο το Ollama, ολόκληρο το pipeline — LLM, εκτέλεση εργαλείων, dataframe — τρέχει on-device. Μηδενική έξοδος δεδομένων. Μηδενική επιβάρυνση GDPR. *Trade-off που έγινε αποδεκτό*: απαιτεί ικανό τοπικό hardware και ένα running instance του Ollama.

ΠΡΟΚΛΗΣΗ Τα ChatGPT ADA, Hex, και Julius AI δείχνουν όλα κώδικα στην έξοδό τους. Τη στιγμή που εμφανίζεται κώδικας, ένα σημαντικό ποσοστό των target χρηστών κλείνει το tab.

ΛΥΣΗ ✓ Το Gnosis είναι αρχιτεκτονικά απαγορευμένο να δείχνει οτιδήποτε τεχνικό. Χωρίς Python. Χωρίς SQL. Χωρίς config. Το tool badge (`run_random_forest`) λέει στον χρήστη τι έτρεξε· δεν δείχνει ποτέ το πώς. *Trade-off που έγινε αποδεκτό*: οι power users που θέλουν να επιθεωρήσουν ή να τροποποιήσουν την ανάλυση δεν μπορούν — θα πρέπει να χρησιμοποιήσουν το Hex, το Julius Pro, ή ένα notebook.

06 ΑΝΤΑΓΩΝΙΣΤΙΚΟ ΤΟΠΙΟ

> **Ειλικρινής τοποθέτηση:** Το Gnosis AI κερδίζει όταν ο χρήστης έχει ένα CSV, μηδενικό λογαριασμό πουθενά, μηδενική επιθυμία να δει κώδικα, και χρειάζεται μια απάντηση στα επόμενα 10 λεπτά. Χάνει όταν χρειάζονται live συνδέσεις βάσης δεδομένων, συνεργασία ομάδας, notebooks με version control, ή enterprise governance. Αυτό είναι σκόπιμο — scope για το v1.

Χαρακτηριστικό	Gnosis AI	ChatGPT ADA	Julius AI	Hex
Δεν απαιτείται λογαριασμός	✓	×	×	×
Δωρεάν, χωρίς όριο μηνυμάτων	✓	$20/mo	$35/mo	$36/editor
Τοπική / offline εκτέλεση	✓ Ollama	×	×	×
Μηδενικός κώδικας στην έξοδο	✓	×	μερικώς	×
Έτοιμο για μη-τεχνικό χρήστη	✓	μερικώς	μερικώς	×
SSE streaming απάντηση	✓	✓	✓	×
Εναλλαγή LLM σε runtime	✓	×	×	×
Συνεργασία ομάδας	×	×	επί πληρωμή	✓
Εκτέλεση οποιουδήποτε κώδικα	×	✓	✓	✓

07 ΤΕΧΝΙΚΗ ΑΡΧΙΤΕΚΤΟΝΙΚΗ

Next.js 14 Frontend · FastAPI Backend · LangChain RAG · Docker · CI/CD

Επίπεδο	Λεπτομέρειες
Frontend	Next.js 14 (App Router), React 18, TypeScript · 24 αρχεία πηγαίου κώδικα / ~6.885 γραμμές
Streaming	Server-Sent Events (SSE) μέσω του hook `useStream` με υποστήριξη ακύρωσης · tokens ξεκινούν σε <1s
Backend	FastAPI (Python 3.11) · per-session dataframe store · διαμόρφωση pydantic-settings
RAG / Router	LangChain `ConversationalRetrievalChain` + Chroma vector store + sentence-transformers embeddings
ML Layer	scikit-learn / pandas · registry 13 αλγορίθμων · επεξηγήσεις τύπου SHAP μετά την εκτέλεση
Middleware	Sliding-window rate limiter (ανά session) + ανίχνευση request-ID (`X-Request-ID`)
Deployment	Vercel (Next.js) + HuggingFace Spaces Docker (16GB) · multi-stage build · non-root user · HEALTHCHECK
Export	Δημιουργία PDF (reportlab) · εξαγωγή Markdown (client-side Blob) · διαμοιράσιμα embed links

> **Απομόνωση πακέτου, όχι μονόλιθος:** Το `src/gnosis_analytics` εγκαθίσταται ως ανεξάρτητο πακέτο pip. Η σουίτα tests τρέχει χωρίς να τρέχει ο FastAPI server. Το CLI δουλεύει χωρίς να τρέχει η εφαρμογή Next.js. Τα integration tests κάνουν mock μόνο το network layer — όχι τη business logic. Κάθε επίπεδο είναι ανεξάρτητα testable.

13 Εργαλεία Ανάλυσης — Βασισμένα σε Registry

Εργαλείο	Κατηγορία	Τι κάνει
`run_statistical_analysis`	stats	Περιγραφικά στατιστικά, σύνοψη κατανομής, σημαίες outlier
`run_random_forest`	classification	Classification / regression με γράφημα feature importance
`run_machine_learning`	auto-ml	Γενικός εκτελεστής ML — αυτόματη επιλογή βέλτιστου αλγορίθμου
`run_correlation_analysis`	analysis	Πίνακας Pearson, κορυφαία συσχετιζόμενα ζεύγη
`run_regression`	regression	Γραμμική / λογιστική παλινδρόμηση με συντελεστές
`run_clustering`	unsupervised	K-means / DBSCAN με ανάθεση cluster
`run_time_series`	temporal	Αποσύνθεση, τάση, εποχικότητα
`run_forecast`	temporal	Προβολή προς τα εμπρός από στήλη χρόνου
`run_data_quality_check`	utility	Λείπουσες τιμές, διπλότυπα, αναντιστοιχίες τύπων
`plot_chart`	viz	Προσαρμοσμένο γράφημα από περιγραφή σε φυσική γλώσσα
`clean_data`	utility	Imputation, αφαίρεση διπλοτύπων, εξαναγκασμός τύπου
`generate_pdf_report`	export	Πλήρης αναφορά ανάλυσης σε PDF (reportlab)
`run_async_analysis`	async	Background job με ειδοποίηση webhook + polling

RAG Pipeline · Πώς Λειτουργεί η Επιλογή Εργαλείου

Σημασιολογική δρομολόγηση, όχι αντιστοίχιση λέξεων-κλειδιών — η αρχιτεκτονική που κάνει τα αποτελέσματα ντετερμινιστικά

> **Γιατί RAG εδώ;** Η κλασική χρήση του RAG είναι retrieval-augmented *generation* — ανάκτηση εγγράφων, ένεση στο context, και το LLM γράφει μια απάντηση. Το Gnosis το χρησιμοποιεί διαφορετικά: ως *σημασιολογικό router*. Το vector store κρατάει περιγραφές εργαλείων, όχι δεδομένα. Το LLM δεν βλέπει ποτέ το CSV του χρήστη. Βλέπει μόνο την ερώτηση και μια κατάταξη υποψήφιων εργαλείων, και μετά επιλέγει ένα. Αυτή είναι η αρχιτεκτονική απόφαση που κάνει τις εξόδους ντετερμινιστικές και testable.

Μοντέλο Embedding

`all-MiniLM-L6-v2`

384-διάστατα dense vectors. Τρέχει τοπικά μέσω `sentence-transformers` — μηδενική εξωτερική κλήση API για κάθε απόφαση δρομολόγησης.

Γιατί όχι OpenAI embeddings;
Λανθάνον χρόνος και κόστος. Μια κλήση API embedding σε κάθε μήνυμα χρήστη θα προσθέτει 200–400ms και χρέωση ανά token σε κάθε απόφαση δρομολόγησης. Ένα τοπικό μοντέλο προσθέτει <10ms.

Trade-off:
Χαμηλότερο όριο σε λεπτές σημασιολογικές ομοιότητες — αποδεκτό επειδή οι περιγραφές εργαλείων είναι σύντομες, ελεγχόμενες και μη-αμφίσημες.

Vector Store

Chroma (τοπικό, embedded)

Το Chroma τρέχει in-process ως embedded store — χωρίς ξεχωριστό server, χωρίς network hop, χωρίς infrastructure για διαχείριση. Ξεκινάει μαζί με τη διαδικασία FastAPI.

Γιατί όχι Pinecone / Weaviate;
Τα managed vector stores είναι σωστά όταν έχεις εκατομμύρια έγγραφα ή χρειάζεσαι multi-region replication. Το registry εργαλείων του Gnosis έχει 13 εγγραφές. Ένα managed store θα ήταν λειτουργική πολυπλοκότητα με μηδενικό όφελος.

Trade-off:
Καμία διατήρηση μεταξύ cold restarts (το store ξαναχτίζεται στο startup από το registry). Αποδεκτό — το ξαναχτίσιμο παίρνει <1s.

Λογική Επιλογής Εργαλείου

ConversationalRetrievalChain

Το ερώτημα γίνεται embed → ομοιότητα συνημιτόνου σε σχέση με 13 vectors περιγραφών εργαλείων → επιστρέφονται top-3 υποψήφιοι. Το LLM παίρνει τους υποψήφιους + το ιστορικό συνομιλίας και βγάζει ακριβώς ένα `tool_id`.

Γιατί top-3, όχι top-1;
Η ομοιότητα συνημιτόνου από μόνη της μπορεί να κάνει λάθος κατάταξη όταν τα ερωτήματα είναι αμφίσημα ("ανάλυσε τάσεις" θα μπορούσε να είναι `run_time_series` ή `run_correlation_analysis`). Δίνοντας στο LLM 3 υποψήφιους του επιτρέπει να χρησιμοποιήσει σημασιολογικό context για να αποσαφηνίσει.

Τι γίνεται αν όλοι οι 3 είναι λάθος;
Το LLM καταφεύγει στο

Γιατί έχει σημασία αυτό για την ορθότητα: επειδή το LLM επιλέγει ένα εργαλείο αντί να γράφει κώδικα, κάθε μονοπάτι ανάλυσης είναι μια προ-ελεγμένη συνάρτηση Python με γνωστές εισόδους, γνωστούς τύπους σφαλμάτων, και σταθερό σχήμα εξόδου. Ένα παραισθητικό όνομα στήλης σε κώδικα που παράγεται προκαλεί σιωπηλή κατάρρευση runtime. Μια λάθος επιλογή εργαλείου προκαλεί ένα ανακτήσιμο σφάλμα δρομολόγησης με επισημασμένο κουμπί retry. Ο τρόπος αποτυχίας είναι ορατός και ανακτήσιμος εκ σχεδιασμού.

6 Πάροχοι LLM · Εναλλαγή σε Runtime

– **Groq** — Προεπιλογή. Γρήγορο, δωρεάν. Οικογένεια LLaMA 3. Sub-second P90 σε σύντομα prompts. Απαιτεί μόνο API key στο `.env`.

– **OpenAI** — GPT-4o. Το υψηλότερο όριο ποιότητας. Server key ή BYOK: ο χρήστης επικολλά το δικό του key στο UI· αποθηκεύεται μόνο in-memory· εξαφανίζεται με το TTL της συνεδρίας (2h).

– **Gemini** — Google Gemini 2.5 Flash. Μεγάλο context window. Server key ή BYOK. Εναλλάξιμο σε runtime χωρίς ανανέωση σελίδας.

– **Claude** — Anthropic Claude Sonnet 4.6. BYOK μόνο. Ισχυρό reasoning, ιδανικό για σύνθετα ερωτήματα ανάλυσης. Εναλλάξιμο σε runtime χωρίς ανανέωση σελίδας.

– **Ollama** — Τοπική εκτέλεση. Self-hosted, μηδενική έξοδος δεδομένων. Ολόκληρο το pipeline τρέχει on-device: LLM, εκτέλεση εργαλείων, dataframe. Μηδενικό API key. Μηδενική επιβάρυνση GDPR.

– **Mock** — Ντετερμινιστικές stub απαντήσεις για ανάπτυξη UI και demos. Δεν απαιτείται key — άνοιξε την εφαρμογή και δούλεψε αμέσως.

BYOK · Κύκλος Ζωής Key & Μοντέλο Ασφαλείας

Πώς κινείται το key μέσα στο σύστημα — και πού δεν πηγαίνει ποτέ

Είσοδος

Ο χρήστης επικολλά το key στο πεδίο εισόδου του `Sidebar.tsx`. Στέλνεται μία φορά μέσω HTTPS στο σώμα του αιτήματος προς το `POST /chat`.

Δεν αγγίζει ποτέ υποβολή φόρμας. Δεν εμφανίζεται ποτέ σε URL ή query string. Δεν γράφεται ποτέ στο `localStorage` ή σε οποιαδήποτε αποθήκευση browser — κρατιέται μόνο σε React state για τη διάρκεια της συνεδρίας.

Σε εξέλιξη

Το FastAPI παραλαμβάνει το key στο σώμα του αιτήματος. Περνιέται απευθείας στον client του πάροχου LLM μόνο για αυτό το αίτημα — αποθηκεύεται ως τοπική μεταβλητή, όχι στο αντικείμενο της συνεδρίας.

Δεν γράφεται ποτέ στον δίσκο. Δεν εμφανίζεται ποτέ σε logs του server (η καταγραφή σώματος αιτήματος είναι απενεργοποιημένη για το `/chat`). Δεν αποθηκεύεται προσωρινά μεταξύ αιτημάτων.

Λήξη

TTL συνεδρίας: 2 ώρες. Όταν η συνεδρία λήξει, η εγγραφή στο `SessionStore` εκδιώκεται — το key χάνεται μαζί της.

Το κλείσιμο tab ή η ανανέωση σελίδας καθαρίζει αμέσως το React state. Δεν απαιτείται καμία ενέργεια από την πλευρά του server. Ο operator δεν έχει ποτέ πρόσβαση στο key σε κανένα σημείο αυτού του κύκλου ζωής.

Γιατί υπάρχει αυτό το μοντέλο: ένα δημόσιο deployment χωρίς λογαριασμό μπορεί να προσφέρει ποιότητα GPT-4o ή Gemini χωρίς ο operator να απορροφά το κόστος ανά token για ανώνυμους χρήστες. Ο χρήστης παρέχει το key, ο operator παρέχει την υποδομή. Κανένα από τα δύο μέρη δεν συγκεντρώνει τα credentials του άλλου.

Μετρικές Επίδοσης

Μετρήθηκαν σε deployment HuggingFace Spaces (2 vCPU · 16GB RAM) · 2026

Πάροχος	1ο token P50	1ο token P90	Tokens/s	Δρομολόγηση εργαλείου	Σημειώσεις
Groq (LLaMA 3.1 8B)	<400ms	<1s	~180	~300ms	Προεπιλεγμένος πάροχος
OpenAI (GPT-4o)	~600ms	~1.2s	~80	~500ms	Υψηλότερη ακρίβεια
Gemini (2.5 Flash)	~500ms	~1.1s	~120	~400ms	Μεγάλο context
Ollama (τοπικό, 8B)	~800ms	~2s	~30–60	~600ms	Εξαρτάται από hardware
Mock	<10ms	<10ms	N/A	N/A	Μόνο dev/demo

Καθυστέρηση Embedding

Το `all-MiniLM-L6-v2` τρέχει τοπικά. Το embedding + Chroma cosine search προσθέτει <10ms ανά αίτημα — αδύνατο να γίνει αντιληπτό από τον χρήστη. Σε σύγκριση με το OpenAI `text-embedding-3-small`: 200–400ms ανά κλήση + κόστος API ανά token.

Εκτέλεση εργαλείων ML

Τα εργαλεία scikit-learn τρέχουν σε ξεχωριστό thread (non-blocking). Random Forest σε CSV 10K γραμμών: ~1.2s. Πίνακας συσχέτισης: ~200ms. Async mode (`run_async_analysis`) διαθέσιμο για datasets που υπερβαίνουν το timeout του SSE.

Αποτύπωμα μνήμης

Dataframe store ανά συνεδρία με TTL 2h και LRU eviction στο όριο MAX_SESSIONS. Chroma store: <5MB resident (13 διανύσματα περιγραφής εργαλείων). Μοντέλο `all-MiniLM-L6-v2`: ~90MB στην πρώτη φόρτωση, cached για τη διάρκεια ζωής της διεργασίας.

08 ΑΡΧΙΤΕΚΤΟΝΙΚΗ FRONTEND · 4 ΒΑΣΙΚΑ MODULES

24 αρχεία πηγαίου κώδικα · TypeScript παντού · state ανυψωμένο στη ρίζα — χωρίς Redux, χωρίς Zustand

Sidebar.tsx

1.021 γραμμές · Ανέβασμα dataset + LLM switcher

Ανέβασμα dataset, επιθεωρητής στηλών, LLM switcher + εισαγωγή BYOK key, γρήγορες ενέργειες, εκκίνηση A/B σύγκρισης, αναλυτικά συνεδρίας, εξαγωγή PDF. Το πλουσιότερο component.

Upload LLM Switch BYOK A/B Compare

MessageBubble.tsx

744 γραμμές · Markdown · Σήμα εργαλείου · Επεξηγήσεις ML

Renderer Markdown, σήμα εργαλείου, κάρτα γραφήματος (ResultCard), πάνελ επεξήγησης ML («Γιατί το μοντέλο επέλεξε αυτό;»), επανάληψη σε σφάλμα.

Markdown Charts SHAP Retry

lib/useStream.ts

231 γραμμές · SSE hook με cancel + ανίχνευση sentinel

SSE hook: σύνδεση, προσάρτηση tokens, ανίχνευση sentinels εργαλείων (`___tool:X___`), ακύρωση, retry με timeout σιωπής. Τυλίγει το `EventSource` σε καθαρό interface 60 γραμμών.

SSE EventSource Cancel Sentinel

SmartSuggestions

Chips context-aware · χωρίς κλήση backend

4 chips προτροπής βαθμονομημένα στο φορτωμένο dataset. Ευρετικοί κανόνες regex ονόματος στήλης (`/surviv|churn|target|label/`) για ανίχνευση δυαδικού στόχου. Παράγονται σύγχρονα — μηδέν αιτήματα δικτύου.

Zero API Heuristics Chips

Σουίτα Tests · 150 Δοκιμές

9 αρχεία δοκιμών · κατώφλι κάλυψης 65% · GitHub Actions CI σε κάθε push

Αρχείο	Δοκιμές	Καλύπτει
`test_pipeline.py`	32	DataCleaner, DataLoader, StatisticsAnalysis, MachineLearningAnalysis, LLM factory, FastAPI routes, vector store
`test_integration.py`	25	End-to-end: health, upload (CSV/Excel/JSON), chat (μονό & πολλαπλών γύρων), PDF download, διαγραφή συνεδρίας
`test_tools.py`	21	Thread-local request context, wrappers ανά εργαλείο, απομόνωση thread υπό ταυτόχρονη εκτέλεση
`test_config.py`	19	pydantic-settings προεπιλογές, επικύρωση, έλεγχοι credentials
`test_run_pipeline.py`	16	Σημείο εισόδου CLI/engine (`run_pipeline()`)
`test_session.py`	13	SessionStore CRUD, λήξη TTL, όριο MAX_SESSIONS, LRU eviction
`test_async_analysis.py`	12	Κύκλος ζωής /analyze/async job, polling κατάστασης, αποστολή webhook
`test_middleware.py`	9	Όρια rate-limit, μετρητές ανά συνεδρία, διάδοση request-ID
`test_compare.py`	3	/compare endpoint: πλήθος γραμμών, κοινές/διαφορετικές στήλες

> **Ειλικρινής σημείωση μηχανικού:** το `tests/test_tools.py` υπάρχει στη σημερινή του μορφή λόγω ενός πραγματικού regression — το thread-local request context έχανε αποτελέσματα ML μεταξύ ταυτόχρονων συνεδριών υπό φορτίο. Η δοκιμή παραμένει στη σουίτα, με το όνομα του bug που εντόπισε. Μια καθαρή λίστα δοκιμών χωρίς ιστορικό είναι ασθενέστερο σήμα από μια δοκιμή που τεκμηριώνει ένα περιστατικό.

09 ΤΕΧΝΙΚΕΣ ΠΡΟΚΛΗΣΕΙΣ

ΠΡΟΚΛΗΣΗ Διαρροή αποτελεσμάτων ML μεταξύ ταυτόχρονων συνεδριών υπό φορτίο — η συνεδρία B έβλεπε στιγμιαία τα αποτελέσματα ML της συνεδρίας A.

ΛΥΣΗ ✓ Το thread-local context χρησιμοποιούσε thread-local storage αλλά δεν επανεκκινούσε μεταξύ αιτημάτων που χειριζόταν το ίδιο worker thread. Διόρθωση: ρητή επανεκκίνηση context στην αρχή κάθε αιτήματος + ειδική δοκιμή regression (`tests/test_tools.py`) που δημιουργεί πραγματικά threads και βεβαιώνει την απομόνωση.

ΠΡΟΚΛΗΣΗ Streaming tokens με σήματα εκκίνησης εργαλείου αναμεμειγμένα στο ίδιο SSE κανάλι — πώς γνωρίζει το frontend πότε ξεκινά ένα εργαλείο;

ΛΥΣΗ ✓ Πρωτόκολλο sentinel: το backend εκπέμπει `__tool:X__` ως ειδικό token πριν ξεκινήσει η αναγνώσιμη απόκριση. Το frontend κάνει regex-match και αφαιρεί τα sentinels πριν την εμφάνιση· τα χρησιμοποιεί μόνο για το σήμα `toolRunning`. Μηδέν ξεχωριστό κανάλι ελέγχου.

ΠΡΟΚΛΗΣΗ Chips context-aware χωρίς κλήση backend σε κάθε φόρτωση dataset — πώς γνωρίζεις ποιες προτάσεις να εμφανίσεις;

ΛΥΣΗ ✓ Τα ονόματα στηλών και οι τύποι δεδομένων επιστρέφονται από το endpoint upload. Ευρετικοί κανόνες regex από την πλευρά του client (`/surviv|churn|target|label|class|fraud/`) εντοπίζουν πιθανούς στόχους. 4 σχετικά chips παράγονται σύγχρονα — μηδέν αιτήματα δικτύου.

ΠΡΟΚΛΗΣΗ Πληκτρολόγιο κινητού που μετατοπίζει το InputBar εκτός οθόνης σε iOS/Android.

ΛΥΣΗ ✓ Το InputBar προσαρτά έναν listener `resize` στο `window.visualViewport`. Όταν το viewport συρρικνώνεται (πληκτρολόγιο ανοιχτό), καλείται το `scrollIntoView({block: 'nearest'})` μέσα σε `requestAnimationFrame` — το textarea παραμένει ορατό πάνω από το πληκτρολόγιο.

ΠΡΟΚΛΗΣΗ Οι Vercel Serverless Functions επιβάλλουν ένα σκληρό, μη διαπραγματεύσιμο όριο 4.5 MB στο request body — τα uploads μεγάλων datasets επέστρεφαν `413` ανεξάρτητα από το δικό μας όριο 50 MB στο backend.

ΛΥΣΗ ✓ Τα uploads παρακάμπτουν εντελώς το Vercel proxy: ο browser στέλνει το αρχείο κατευθείαν στο FastAPI backend, με CORS περιορισμένο στο origin του deployed frontend μέσω `ALLOWED_ORIGIN`, εκτεθειμένο στην πλευρά του client μέσω `NEXT_PUBLIC_BACKEND_URL`. Το chat και τα metadata calls παραμένουν στο proxied route `/api`· άλλαξε μόνο το μονοπάτι μεγάλου payload.

10 ΑΠΟΦΑΣΕΙΣ ΠΡΟΪΟΝΤΟΣ & ΑΝΤΙΣΤΑΘΜΙΣΕΙΣ

ΠΡΟΚΛΗΣΗ **A1 Chat Interface, όχι Φόρμα/Dashboard.** Απορρίφθηκε: form-based query builder με dropdowns για λειτουργία, στήλες, παραμέτρους.

ΛΥΣΗ ✓ Το chat κερδίζει γιατί οι κατηγορίες ενός form builder είναι πεπερασμένες. Η φυσική γλώσσα επιτρέπει στον χρήστη να κάνει ερωτήσεις που ο σχεδιαστής δεν είχε προβλέψει. Ο LLM router χειρίζεται την επίλυση ασάφειας — μηδέν καμπύλη εκμάθησης. *Κόστος αποδεκτό*: ασαφείς ερωτήσεις μπορεί να δρομολογηθούν στο λάθος εργαλείο· μετριάζεται από σήμα εργαλείου + κουμπί επανάληψης.

ΠΡΟΚΛΗΣΗ **A2 Tool Router, όχι Δημιουργία Κώδικα.** Απορρίφθηκε: το LLM παράγει Python/pandas κώδικα on the fly και τον εκτελεί σε sandbox (à la ChatGPT ADA).

ΛΥΣΗ ✓ Το tool routing κερδίζει γιατί ο παραγόμενος κώδικας αποτυγχάνει με απρόβλεπτους τρόπους — ψευδαισθητικά ονόματα στηλών, λανθασμένες κλήσεις API, σφάλματα import. Ένα προδοκιμασμένο εργαλείο είτε εκτελείται σωστά είτε επιστρέφει γνωστό τύπο σφάλματος. Η εκτέλεση ξεκινά άμεσα — χωρίς επιπλέον γύρο LLM. *Κόστος αποδεκτό*: ασυνήθης ανάλυση εκτός καταλόγου απαιτεί εξαγωγή σε notebook.

ΠΡΟΚΛΗΣΗ **A3 SSE Streaming, όχι WebSocket.** Απορρίφθηκε: WebSocket για αμφίδρομη επικοινωνία πραγματικού χρόνου.

ΛΥΣΗ ✓ Το SSE κερδίζει γιατί το μοτίβο επικοινωνίας είναι αυστηρά μονοκατευθυντικό: ο client στέλνει, ο server κάνει stream. Το SSE τρέχει πάνω από απλό HTTP/2, επανασυνδέεται αυτόματα μέσω του browser's `EventSource` API, χωρίς απαιτούμενη χειραψία αναβάθμισης. Η αμφίδρομη φύση του WebSocket είναι overhead χωρίς όφελος. *Κόστος αποδεκτό*: ο server δεν μπορεί να εκκινήσει μηνύματα χωρίς αίτημα client — δεν αποτελεί περιορισμό για αυτή τη χρήση.

Αυτόνομο CLI

Ίδια analytics engine · 3 σημεία εισόδου: Web App / Async Jobs / Terminal

```
# Εγκατάσταση engine ως pip package pip install -e ./src
# Εκτέλεση πλήρους ML ανάλυσης από τερματικό gnosis analyze  --file
data.csv  --mode ml  --target Survived  --output report.pdf
# Modes: stats | ml | all # Τα Algorithm IDs ταυτίζονται με το registry που
χρησιμοποιεί το API
```

> Το ίδιο δοκιμασμένο μονοπάτι κώδικα εκτελείται και στα τρία πλαίσια. Δεν υπάρχει «έκδοση CLI» έναντι «έκδοσης API». Ένα bug που διορθώνεται στο engine διορθώνεται παντού. Ένα εύρημα από το chat UI μπορεί να αναπαραχθεί από το τερματικό με μια εντολή.

11 ΟΔΙΚΟΣ ΧΑΡΤΗΣ

Παραδόθηκε (v1)

✓ Ζωντανή ανάπτυξη σε παραγωγή (Vercel + Hugging Face Spaces)

✓ Async ανάλυση με ειδοποίηση webhook

✓ Server-side A/B σύγκριση με στατιστική διαφορά ανά στήλη

✓ Αυτόνομο CLI wrapper

✓ CI pipeline με αναφορά κάλυψης Codecov

✓ Multi-stage Docker build + deployment σε HuggingFace Spaces

✓ 150 δοκιμές, κατώφλι κάλυψης 65%

✓ 5 LLM providers με εναλλαγή σε runtime

✓ Μοντέλο BYOK key (μόνο in-memory)

Σε Εξέλιξη

→ Phase 2 Panel: clustering, μείωση διαστάσεων (imports sidebar ήδη στον κώδικα)

→ Διευρυμένη κάλυψη LLM: HuggingFace Inference (factory ήδη επεκτάσιμο)

→ CI matrix Python 3.9–3.11 για backwards compatibility

Σήματα v2

◇ Κοινή χρήση με authentication: ονομαστές συνεδρίες, μόνιμοι χώροι εργασίας

◇ Κατάλογος εργαλείων plugin: νέα εργαλεία απαιτούν μόνο backend + μία εγγραφή registry

◇ Connectors βάσεων δεδομένων: Postgres, Snowflake

◇ Έλεγχοι πρόσβασης ομάδας και κοινόχρηστες συνεδρίες

12 ΑΝΑΛΟΓΙΣΜΟΙ & ΔΙΔΑΓΜΑΤΑ

Έχτισα το Gnosis AI για να απαντήσω σε ένα ερώτημα που συναντούσα επανειλημμένα: γιατί η διερευνητική ανάλυση δεδομένων εξακολουθεί να απαιτεί γνώση Python ή συνδρομή σε BI, όταν η ερώτηση που θέλει να κάνει ο χρήστης είναι συχνά απλή;

Η αρχιτεκτονική απόφαση που ορίζει το προϊόν — *δρομολόγηση εργαλείων αντί για παραγωγή κώδικα* — δεν ήταν η πρώτη ιδέα. Αναδύθηκε μετά από 7 συνεντεύξεις, μετά από ένα πρώτο prototype που εμφάνιζε Python κώδικα στον χρήστη και τον έχανε, και μετά από το bug του thread-local context που αποκάλυψε πού βρισκόταν η πραγματική πολυπλοκότητα.

Κάθε feature υπάρχει γιατί αφαιρεί τριβή από μια συγκεκριμένη στιγμή του χρήστη — όχι γιατί ήταν τεχνικά ενδιαφέρον να κατασκευαστεί.

Τι θα έκανα διαφορετικά: θα ξεκινούσα νωρίτερα με το πρόβλημα απομόνωσης συνεδρίας. Το bug του thread-local context εντοπίστηκε από μια δοκιμή — αλλά αυτή η δοκιμή γράφτηκε *μετά* την εμφάνιση του bug υπό φορτίο. Ένα μοντέλο ταυτόχρονης εκτέλεσης (ένα dataframe store ανά αίτημα, όχι ανά συνεδρία) θα ήταν η καθαρότερη προεπιλογή

από την αρχή.

Spilios Dimakopoulos · **github.com/SpiliosDimakopoulos/gnosis-ai** · **Next.js 14 · FastAPI · LangChain RAG · 2026**