CS Tutor AI - Complete Project Documentation

# Project Overview

CS Tutor AI is an AI-powered coding assistant that enhances learning experiences by explaining code snippets and providing interactive educational features. Using Streamlit as our frontend, we create a web- based interface that allows users to generate, understand, and learn from AI-assisted code explanations.

**Project Deadline**: May 12, 2025

# Key Features

1. **Intelligent Code Generation**: Standard code completion and generation
2. **Explanation Mode**: Detailed breakdown of code functionality, time/space complexity, and design choices
3. **Learning Mode**: Interactive tutoring experience with adjustable difficulty levels
4. **Language-Specific Best Practices**: Customized guidance for different programming languages
5. **Contextual Awareness**: Adapts explanations based on the user's skill level and project context

# Technical Architecture

## System Components

┌┌──────────────────────────────────────────┐┐ ┌┌──────────────────────────────────────────┐┐

││ SSttrreeaammlliitt

AApppp

││◄◄────────►►││

BBaacckkeenndd

SSeerrvviiccee ││

└└──────────────────────────────────────────┘┘ └└──────────────────┬┬──────────────────────┘┘

││

┌┌────────────────▼▼──────────────────────┐┐

││ FFiinnee--ttuunneedd LLLLMM ││

└└────────────────────────────────────────┘┘

## Tech Stack

### AI/ML Components

 Base Model: CodeLlama-34B or StarCoder2-15B  Fine-tuning: QLoRA/LoRA using PEFT library

 Inference Engine: vLLM or Text Generation Inference (TGI)  Quantization: GPTQ or AWQ for performance optimization

### Backend

 API Framework: FastAPI

 Deployment: Docker containers

 Database: SQLite for session history (optional)

### Frontend

 Streamlit for web interface

 Streamlit components for syntax highlighting and interactive elements  Communication: HTTP requests to backend

# Development Roadmap

## Day 1-2: Foundation & Setup (May 9-10)

 Initialize GitHub repository

 Set up development environment

 Define project architecture and components  Set up Streamlit app skeleton

 Select base model for backend  Create initial documentation

## Day 3: Backend Development (May 11)

 Implement FastAPI backend endpoints  Set up model inference pipeline

 Create API for code generation and explanation  Test API functionality

 Connect to pre-trained model (use existing model rather than fine-tuning to save time)

## Day 4: Frontend & Integration (May 12)

 Build Streamlit UI components

 Implement code editor with syntax highlighting  Create explanation display panels

 Set up user preferences (language, difficulty level)  Connect frontend to backend API

 Perform end-to-end testing  Deploy MVP

 Complete final documentation

# Technical Implementation Details

## Streamlit App Structure

ccss--ttuuttoorr--ssttrreeaammlliitt//

├├────

├├────

aapppp..ppyy ## ppaaggeess//

MMaaiinn

SSttrreeaammlliitt

aapppplliiccaattiioonn

││ ├├────

hhoommee..ppyy

## HHoommee//llaannddiinngg

ppaaggee

││ ├├────

ggeenneerraattee..ppyy

## CCooddee

ggeenneerraattiioonn

ppaaggee

││ ├├────

eexxppllaaiinn..ppyy

## CCooddee

eexxppllaannaattiioonn

ppaaggee

││ └└────

lleeaarrnn..ppyy

## IInntteerraaccttiivvee

lleeaarrnniinngg

ppaaggee

├├──── ccoommppoonneennttss//

││ ├├────

ccooddee

eeddiittoorr..ppyy

## CCuussttoomm

ccooddee

eeddiittoorr

ccoommppoonneenntt

││ ├├────

eexxppllaannaattiioonn..ppyy

## EExxppllaannaattiioonn

ddiissppllaayy

ccoommppoonneenntt

││ └└────

sseettttiinnggss..ppyy

## UUsseerr

sseettttiinnggss

ccoommppoonneenntt

├├──── uuttiillss//

││ ├├────

aappii

cclliieenntt..ppyy

## BBaacckkeenndd

AAPPII

ccoommmmuunniiccaattiioonn

││ └└────

ccooddee

ffoorrmmaatttteerr..ppyy ##

CCooddee

ffoorrmmaattttiinngg

uuttiilliittiieess

├├────

└└────

ccoonnffiigg..ppyy ## rreeqquuiirreemmeennttss..ttxxtt ##

CCoonnffiigguurraattiioonn DDeeppeennddeenncciieess

sseettttiinnggss

## Backend API Endpoints

PPOOSSTT //aappii//ggeenneerraattee

-- GGeenneerraatteess

ccooddee bbaasseedd

oonn pprroommpptt

aanndd ccoonntteexxtt

-- RReettuurrnnss

ggeenneerraatteedd

ccooddee

PPOOSSTT //aappii//eexxppllaaiinn

-- TTaakkeess ccooddee

aanndd

ggeenneerraatteess eexxppllaannaattiioonnss

-- RReettuurrnnss

ddeettaaiilleedd

eexxppllaannaattiioonn

aatt rreeqquueesstteedd ddeepptthh

PPOOSSTT //aappii//iinntteerraaccttiivvee

-- HHaannddlleess

ffoollllooww--uupp qquueessttiioonnss

aabboouutt

ccooddee

-- RReettuurrnnss

aannsswweerrss

ttoo ssppeecciiffiicc qquueessttiioonnss

GGEETT //aappii//ssttaattuuss

-- RReettuurrnnss

mmooddeell

aanndd

sseerrvviiccee ssttaattuuss

## User Interaction Flow

1. User visits the CS Tutor AI Streamlit app
2. User selects desired mode (generate, explain, learn)
3. For code generation:

 User enters a description of the code they need  App sends request to backend

 Generated code is displayed with syntax highlighting

1. For code explanation:

 User pastes/writes code in the editor

 Selects explanation depth (basic, intermediate, advanced)  App sends code to backend

 Detailed explanation is displayed in panels

1. For learning mode:

 User selects a topic and difficulty level

 App provides guided learning experience with code examples and explanations  User can ask follow-up questions

## Model Prompt Template

python

PPRROOMMPPTTTTEMPLATEETALPME == """"""

MMooddee:: {{mmooddee}}

LLaanngguuaaggee:: {{llaanngguuaaggee}}

DDiiffffiiccuullttyy:: {{ddiiffffiiccuullttyy}}

UUsseerr LLeevveell:: {{userresu llevel}}leve

CCooddee::

{{ccooddee}}

QQuueessttiioonn//TTaasskk::

{{qquueessttiioonn}}

IInnssttrruuccttiioonnss::

{{iinnssttrruuccttiioonnss}} """"""

# Implementation Strategy

## Backend Implementation (FastAPI)

python

*## EExxaammppllee FFastAPIIPAtsa BBackenddnekca CCodeedo*

ffrroomm ffastapiipatsa iimporttropm FFastAPI,,IPAtsa BBodyydo

ffrroomm ppydanticcitnady iimporttropm BBaseModelledoMesa

iimmppoorrtt uuvicornnrociv iimmppoorrtt ooss

ffrroomm ttransformerssremrofsnar iimporttropm ppipelineenilepi

aapppp == FFaassttAAPPII(())

*## IInniittiiaalliizzee mmodelledo ((useesu ppre-traineddeniart-er ttoo ssaveeva ttime))emi*

mmooddeell == ppiippeelliinnee((

""tteexxtt--ggeenneerraattiioonn"",, mmooddeell==""ccooddeellllaammaa//CCooddeeLLllaammaa--77bb--IInnssttrruucctt--hhff"",,

ddeevviicceemmap="auto""otua"=pa

))

ccllaassss CCooddeeRReeqquueesstt((BBaasseeMMooddeell))::

ccooddee:: sstrrt == """"

llaanngguuaaggee:: ssttrr == ""ppyytthhoonn""

mmooddee:: sstrrt == ""explain""nialpxe

ddiiffffiiccuullttyy:: ssttrr == ""iinntteerrmmeeddiiaattee""

qquueessttiioonn:: sstrrt == """"

@@aapppp..ppoosstt((""//aappii//eexxppllaaiinn""))

ddeeff eexxppllaaiinnccode(request::tseuqer(edo CCodeRequest)::)tseuqeRedo

pprroommpptt == ff""""""

EExxppllaaiinn ttheeh ffollowinggniwollo {{request.language}}egaugnal.tseuqer ccodeedo iinn {{request.difficulty}}ytluciffid.tseuqer llevelleve tterms::smre

``````{{rreeqquueesstt..llaanngguuaaggee}}

{{rreeqquueesstt..ccooddee}}

``````

PPrroovviiddee aa ddetaileddeliate eexplanationnoitanalpx ccovering::gnirevo

11.. WWhhaatt ttheeh ccodeedo ddoesseo

22.. HHooww iitt wworksskro ssteppet bbyy ssteppet

33.. TTiimmee aanndd sspaceecap ccomplexityytixelpmo

44.. BBeesstt pprraaccttiicceess aanddn ppotentiallaitneto iimprovementsstnemevorpm """"""

rreessppoonnssee == mmooddeell((pprroommpptt,, mmaxxa llength=1024,,4201=htgne ttemperature=0.7))7.0=erutarepme

rreettuurrnn {{"explanation"::"noitanalpxe" rresponse[0]["generateddetareneg"[]0[esnopse ttext"]}}]"txe

*## AAdddd ootthheerr*

*eennddppooiinnttss::*

*ggeenneerraattee,, iinntteerraaccttiivvee,,*

*eettcc..*

iiff

nnaammee ====

"" mmaaiinn ""::

uuvviiccoorrnn..rruunn((aapppp,, hhoosstt==""00..00..00..00"",, ppoorrtt==88000000))

## Streamlit App Implementation

python

*## EExxaammppllee mmaaiinn aapppp..ppyy*

iimmppoorrtt sstreamlittilmaert aass sstt iimmppoorrtt rrequestsstseuqe

ffrroomm ccomponents.codeedoc.stnenopmo eeditorrotid iimporttropm ccreateetaer ccodeedo eeditorrotid

ffrroomm ccomponents.explanationnoitanalpxe.stnenopmo iimporttropm ccreateetaer eexplanationnoitanalpx ppanellena

ffrroomm uutils.apiipa.slit cclienttneil iimporttropm AAPIClienttneilCIP

sstt..sseettppaaggeeccoonnffiigg((ppaaggeettiittllee==""CCSS TTuuttoorr AAI",,"I llayout="wide"))"ediw"=tuoya

sstt..ttiittllee((""CCSS TTuuttoorr AAII -- YYourruo IIntelligenttnegilletn CCodinggnido AAssistant"))"tnatsiss

*## IInniittiiaalliizzee AAPIIP cclienttneil*

aappiicclliieenntt == AAPPIICClliieenntt((bbaasseeuurrll==""hhttttpp::////llooccaallhhoosstt::88000000""))

*## MMooddee sseelleeccttiioonn*

mmooddee == sstt..ssiiddeebbaarr..sseelleeccttbbooxx((

""SSeelleecctt MMooddee::"",,

[[""CCooddee GGeenneerraattiioonn"",, ""CodeedoC EExplanation",,"noitanalpx ""InteractiveevitcaretnI LLearning"]]"gninrae

))

*## LLaanngguuaaggee sseelleeccttiioonn*

llaanngguuaaggee == sstt..ssiiddeebbaarr..sseelleeccttbbooxx(( ""PPrrooggrraammmmiinngg LLaanngguuaaggee::"",, [[""PPyytthhoonn"",, ""JJaavvaaSSccrriipptt"",, ""JJaavvaa"",,

""CC++++"",,

""GGoo""]]

))

*## DDiiffffiiccuullttyy//ddeettaaiill lleevveell*

ddiiffffiiccuullttyy == sstt..ssiiddeebbaarr..sseelleeccttsslliiddeerr(( ""EExxppllaannaattiioonn DDeettaaiill::"",,

ooppttiioonnss==[[""BBaassiicc"",, ""Intermediate",,"etaidemretnI ""Advanced"]]"decnavdA

))

*## MMaaiinn iinterfaceecafretn bbaseddesa oonn mmodeedo*

iiff mmodeedo ==== ""CodeedoC EExplanation"::"noitanalpx

sstt..hheeaaddeerr((""CCooddee EExxppllaannaattiioonn""))

*## CCooddee iinnppuutt*

ccooddee == ccreateetaer ccodeedo eeditor(language.lower())))(rewol.egaugnal(rotid

iiff sst.button("ExplainnialpxE"(nottub.t CCode")::)"edo

iiff ccode::edo

wwithhti sst.spinner("GeneratinggnitareneG"(rennips.t eexplanation...")::)"...noitanalpx

rreessppoonnssee == aappiicclliieenntt..eexxppllaaiinnccode((edo ccooddee==ccooddee,,

llaanngguuaaggee==llaanngguuaaggee..lloowweerr(()),,

ddiiffffiiccuullttyy==ddiiffffiiccuullttyy..lloowweerr(())

))

*## DDiissppllaayy eexxppllaannaattiioonn*

ccrreeaatteeeexxppllaannaattiioonnppanel(response["explanation"]))]"noitanalpxe"[esnopser(lena eellssee::

sstt..wwaarrnniinngg((""PPlleeaassee eenterretn ssomeemo ccodeedo ttoo eexplain."))".nialpx

*## AAdddd ootthheerr*

*mmooddeess::*

*ggeenneerraattiioonn,,*

*lleeaarrnniinngg,,*

*eettcc..*

# Evaluation Metrics

1. **Code Correctness**: % of generated code that runs without errors
2. **Explanation Quality**: Rated by test users (1-5 scale)
3. **Response Time**: Latency for generating responses
4. **Learning Effectiveness**: User-reported understanding improvement

# Advantages of Streamlit Approach

1. **Rapid Development**: Streamlit allows for quick iteration and prototyping
2. **Web Accessibility**: Users can access via any browser without installing extensions
3. **Simple Deployment**: Can be deployed to Streamlit Cloud or any cloud service
4. **Rich UI Components**: Streamlit provides many built-in components for interactive interfaces
5. **Easy Demo**: Ideal for showcasing the project's functionality

# Deployment Options

### Local Development:

streamlit run app.py

1. **Streamlit Cloud**: Push to GitHub and deploy directly from Streamlit Cloud
2. **Docker Containers**: Package both frontend and backend in containers
3. **Cloud Services**: Deploy on AWS, GCP, or Azure

# Future Enhancements

1. **Multimodal Support**: Add visualization for algorithms and data structures
2. **Progress Tracking**: Monitor user learning and adapt difficulty
3. **Custom Domain Support**: Allow training on company-specific codebases
4. **Offline Mode**: Enable fully local operation
5. **VS Code Extension**: Expand to IDE integration as a future enhancement

# Required Skills

 Python development

 Streamlit UI design

 FastAPI backend development

 AI/ML knowledge (transformers)

# Resource Requirements

 Development machine

 GPU for inference (or cloud-based GPU service)  Storage for models

 Deployment server (or cloud service)

# Testing Strategy

1. Unit tests for backend endpoints
2. Integration tests for API communication
3. User acceptance testing with CS students

# Deployment Checklist

 All planned features implemented  Documentation complete

 Performance metrics meet targets  Security review completed

 Deployment configuration ready

 README and user guides finalized

This documentation is a living document and will be updated throughout the development process.