Criterion B: Design

Design Overview

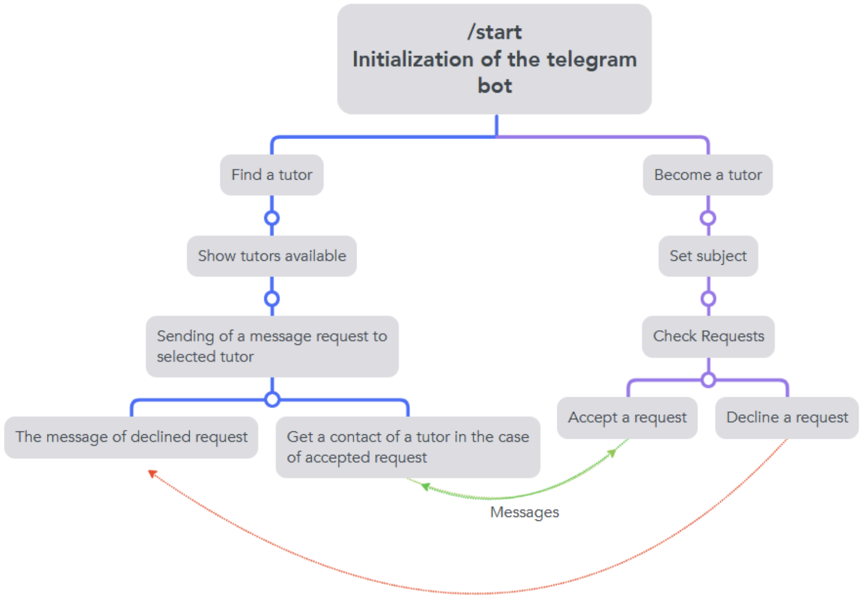
System Architecture

1. **Client-Side**:
    - **Platform**: Telegram app (mobile/desktop) for students and tutors.
    - **Interface**: Text-based interaction with buttons and menus (no GUI installation required).
    - **Interaction**: Uses inline keyboards for actions like accepting requests and text-based messaging.
    - **User Roles**:
        - **Students**: Search for tutors, and send requests.
        - **Tutors**: Set subjects, manage availability, and respond to requests.
2. **Server-Side**:
    - **Backend**: Python script hosted on a server (e.g., Heroku or Replit) using `pyTelegramBotAPI`.
    - **Key Functions**:
        - Process user commands (e.g., `/start`, `Find a tutor`).
        - Handle database queries (fetch tutors, update availability).
        - Manage real-time notifications (e.g., request confirmations).
3. **Database**:
    - **SQLite**: Serverless relational database stored in a local file (`tutors.db`).
    - **Tables**:
        - **Users**: Stores user profiles (ID, username, mode, subject).

- **Orders**: Tracks tutoring requests (sender, recipient, message, status).

Flowchart of User Interaction

A simplified flowchart is provided below:


/start
Initialization of the telegram bot
Find a tutor
Show tutors available
Sending of a message request to selected tutor
The message of declined request
Get a contact of a tutor in the case of accepted request
Become a tutor
Set subject
Check Requests
Accept a request
Decline a request
Messages

1. **Wireframe of Bot Commands**

Component	Description
Main menu	Buttons: "Find a tutor", "Become a tutor".
Tutor Setup	"Set Subject" → Text input for a subject.
Request System	Inline buttons: "Accept/Decline" with message threading.
Messaging	"📩 Text back" and "📩 Reply" buttons for ongoing conversations.

2. **Database Schema**
    - **Users table:**

Field	Data Type	Description
id	INTEGER	Telegram user ID (Primary Key).
username	TEXT	Telegram handle (e.g., @arlan_suleimanov).
mode	TEXT	Find (for a student) or Become (for a tutor).

subject	TEXT	Tutor’s subject (e.g., "Physics SL").

- **Orders table:**

Field	Data Type	Description
id	INTEGER	Unique request ID (Autoincrement).
sender_id	INTEGER	Student’s Telegram ID.
recipient_id	INTEGER	Tutor’s Telegram ID.
message	TEXT	Initial request text.
status	TEXT	Pending/Accepted/Declined.

3. **Error Handling Plan**

Scenario	Handling Method	User Feedback
Invalid subject input	Free-text validation (no restricted subjects).	Auto-saved; tutor can update it later.
Tutor not found	Check mode = 'Become' in the Users table.	"No tutors available. Try later!"

Message send failure	Catch Exception in send_to_student()/send_to_tutor.	"Failed to send. User may have blocked the bot."
Duplicate tutor registration	mode is reset to NULL on "/start".	Users can re-register without conflicts.