Feature	Description	Comment
M1: Secure Compute		
Design	Write design document with expert peer review, including architecture and threat models.	See here: https://github.com/projectglove/glove-monorepo/blob/main/README.md
Enclave planning and feasibility	Determine the best confidential compute service provider to use. Write an example enclave.	See here: https://github.com/projectglove/glove-monorepo/blob/main/README.md
Enclave PoC	Write a simple PoC, which nets a list of input numbers. The most important objective being to get remote attestation working so that we can demonstrate: How to produce the open source image – that the enclave is using – from a git repo a. That the built open source image fingerprint matches the remotely attested image fingerprint. b. That we can feasibly show in an easy-to-interpret method that the enclave is running the code we claim it to be running	See here: https://github.com/projectglove/glove-monorepo/commit/f551045a4d980feb44c6423d783b05042e289215
Secure comms	Add secure communication between the enclave and the client. The communication between the client and a load balancer happens over the HTTPS protocol, and from LB to the VM hosting enclave over private VPC dedicated to glove only.	See here: https://github.com/projectglove/glove-monorepo/blob/main/docs/technical-design.md#voting-workflow
Algorithm	Implement the vote netting algorithm and ongoing training model.	See here:

		https://github.com/projectglove/glove-monorepo/commit/d8c29df5d920668fab4c5a4eaf4995b449fabfb6
Deployment and operations	Deploy to Rocco and Kusama with instructions	See here: https://github.com/projectglove/glove-monorepo?tab=readme-ov-file#deployment
M2: Proxy key management		
Evaluate	Evaluate potential options	See output here: https://github.com/projectglove/glove-monorepo/blob/main/docs/technical-design.md
Implementation	Implement chosen option	See here: https://github.com/projectglove/glove-monorepo/blob/main/docs/technical-design.md#cryptographic-keys
Deployment and operations	Deploy with instructions of how to do so	https://github.com/projectglove/glove-monorepo?tab=readme-ov-file#running-the-glove-service
M3: Data collection		
Subsquid	Write a Subsquid which collects Opengov data for use on the front-end.	Glove backend service uses Subscan moderately to query votes, blocks, extrinsics, and is also used in the vote and verify-vote methods: https://github.com/projectglove/glove-monorepo/blob/main/client/src/lib.rs

		In a concerted effort to have one source of truth, fronted dApp also uses Subscan to query active treasury referenda for the main referendum list and mixed vote data submitted by the Glove proxy: https://github.com/projectglove/frontend/blob/main/lib/utils.ts
Deployment and operations	Deploy the subsquid	SLA and continuous maintenance of endpoints are provided by Subscan; Glove queries their endpoints directly which minimises the need for a private node.
M4 Back-end		
Data Model	Design the data model for the necessary data used by the application	See here: https://github.com/projectglove/glove-monorepo/blob/main/docs/technical-design.md#glove-proof-generation
Enclave comms	Implement secure message between user and enclave, whereby the back-end relays messages between enclave and user.	Implementation based on: https://github.com/projectglove/glove-monorepo/blob/main/docs/technical-design.md#voting-workflow
API	Implement the API for the front-end	See here: https://github.com/projectglove/glove-monorepo?tab=readme-ov-file#rest-api
Deployment and operations	Deployment with instructions	See here: https://github.com/projectglove/glove-monorepo?tab=readme-ov-file#deployment

M5: Front-end		
Requirements gathering	Conduct extensive user testing to easily facilitate anonymous voter participation	Reflected in front-end repo
Wireframes	Initial concepts created using wireframing tools	See here: https://github.com/projectglove/frontend/blob/main/Glove%20Wireframes.pdf
Mockups	Created using Google slides	See here: https://github.com/projectglove/frontend/tree/main/mockups
HTML and CSS	HTML and Tailwind CSS scaffolding via Nextjs	See here: https://github.com/projectglove/frontend/blob/main/package.json
Web components	Made with React and Typescript	See here: https://github.com/projectglove/frontend/tree/main/components
Wallet Integrations	Integrated Polkadot JS and Talisman via Polkadot injected extensions (extension-dapp)	See here: https://github.com/projectglove/frontend/blob/main/components/connect-wallet.tsx
Deployment	Development and production frontend deployments provided by Vercel	Glove is a Nextjs app, and Vercel is a built-in deployment tool for frontend development created by the same makers of Nextjs, so it's already part of the dev pipeline. Installation instructions (includes Kusama deployment): https://github.com/projectglove/frontend?tab=

		readme-ov-file#getting-started
M6: System testing		
Pre-Production UAT	End-to-end user testing provided by Cypress, unit testing provided by Jest.	Unit tests: https://github.com/projectglove/frontend/tree/main/components/__tests__/unit E2E tests: https://github.com/projectglove/frontend/tree/main/cypress Incorporated here: https://github.com/projectglove/glove-monorepo/blob/main/build.sh Deprecation of Rococo Testnet after delivery of M1, M2, M3, M4 and M5 requires project team to amend and reimplement certain aspects of testing suite
Threat Modeling Review	Threat Modeling Review	See here: https://github.com/projectglove/glove-monorepo/blob/main/docs/technical-design.md#threat-model
Security and Penetration Testing	Security and Penetration Testing	Completed extensive internal testing with support from enclave experts
Secure Compute Simulations	Secure Compute Simulations	Completed extensive internal testing with support from enclave experts

Patches, upgrades, enhancements	Patches, upgrades, enhancements	Various, including: https://github.com/projectglove/glove-monorepo/pull/47